



Network Programming

JAVA Network Programming

Network Programming

- **Network Programming** involves writing programs that communicate with other programs across a computer network.

Network Programming

- There are many issues that arise when doing network programming.
 - which do not appear when doing single program applications.
- However, **JAVA** makes networking applications simple.
 - due to the easy-to-use libraries.

Network Programming

- In general, applications that have components running on different machines are known as **distributed applications**.
 - usually they consist of client/server relationships.

client/server

- A **server** is an application that **provides** a "service" to various **clients** who **request** the service.
- There are many client/server scenarios in real life.

peer-to-peer

- In the **general** networking scenario, everybody can either be a client or a server at any time.
 - This is known as **peer-to-peer** computing.
- In terms of writing **java applications** it is similar to having **many applications** communicating among one another.

communication between applications in JAVA

- There are many different strategies for allowing communication between applications.

communication between applications in JAVA

➤ JAVA technology allows:

- internet clients to connect to servlets or back-end business systems (or databases).
- applications to connect to one another using sockets.
- applications to connect to one another using RMI (remote method invocation).
- some others.

communication between applications in JAVA

We will look at the simplest strategy of connecting applications using **sockets.**

communication between applications in JAVA

A **Protocol** is a standard pattern of **exchanging** information.

communication Protocol

- There are layered **levels** of protocols.
 - in that some **low level** layers deal with how to transfer the **data bits**
 - others deal with more **higher-level** issues such as "**where to send the data to**".

high-level Application Layer protocols

- Computers running on the internet typically use one of the following high-level **Application Layer protocols** to allow applications to communicate:
 - Hyper Text Transfer Protocol (HTTP)
 - File Transfer Protocol (FTP)
 - Telnet

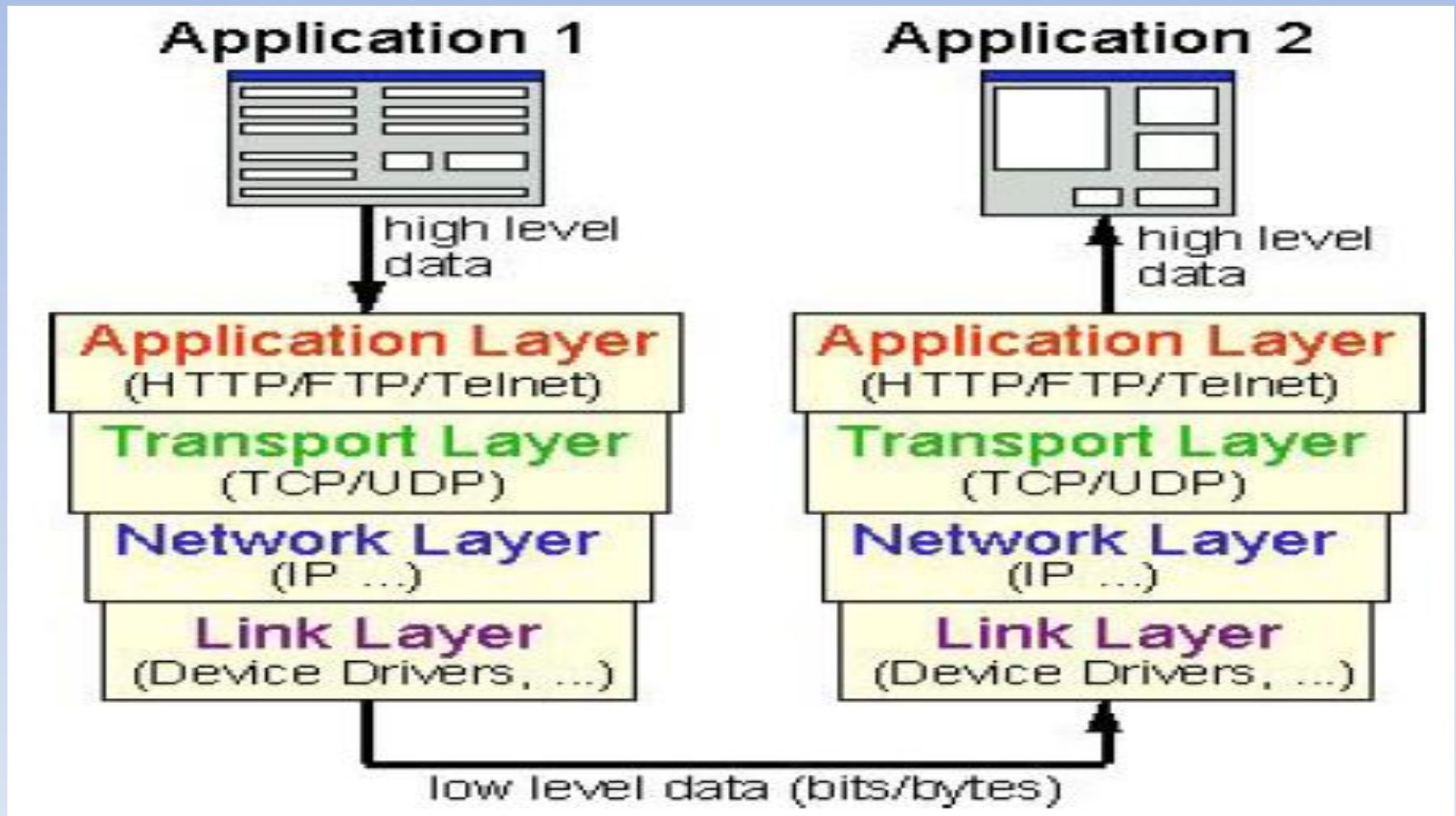
Transport Layer protocols

- Transport Layer protocols determine how the data is to be transported from one machine to another:
 - Transport Control Protocol (TCP)
 - User Datagram Protocol (UDP)

Network and Links Layer

- A **Network Layer** is determining how to locate destinations for the data (i.e., address).
- The lowest level (for computers) there is a **Link Layer** which actually handles the transferring of bits/bytes.

communication between applications



communication between applications in JAVA

- When you write **JAVA applications** that communicate over a network, you are programming in the **Application Layer**.
- JAVA allows two types of communication via two main types of Transport Layer protocols.
 - (TCP) and (UDP) protocols

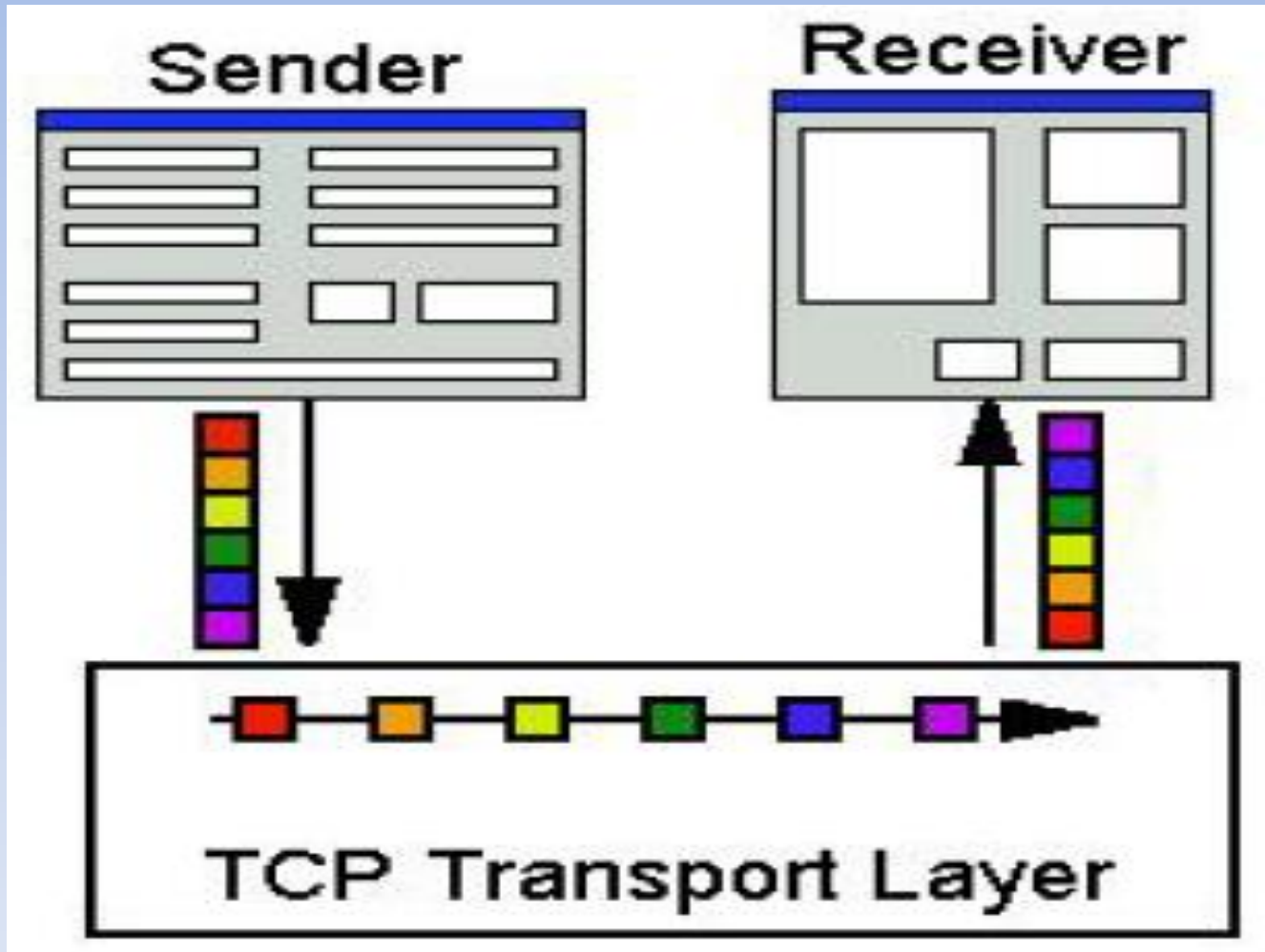
Transport Control Protocol (TCP)

- a **connection-based** protocol that provides a **reliable** flow of data between two computers.
- guarantees that data sent from one end of the connection actually gets to the other end and in the **same order**
 - similar to a phone call.
 - Your words come out in the order that you say them.

Transport Control Protocol (TCP)

- provides a **point-to-point** channel for applications that require reliable communications.
- **slow overhead** time of setting up an end-to-end connection.

Transport Control Protocol (TCP)



User Datagram Protocol (UDP)

- a protocol that sends **independent packets** of data, called **datagrams**, from one computer to another.
- no guarantees about arrival.
 - UDP is not connection based like TCP.

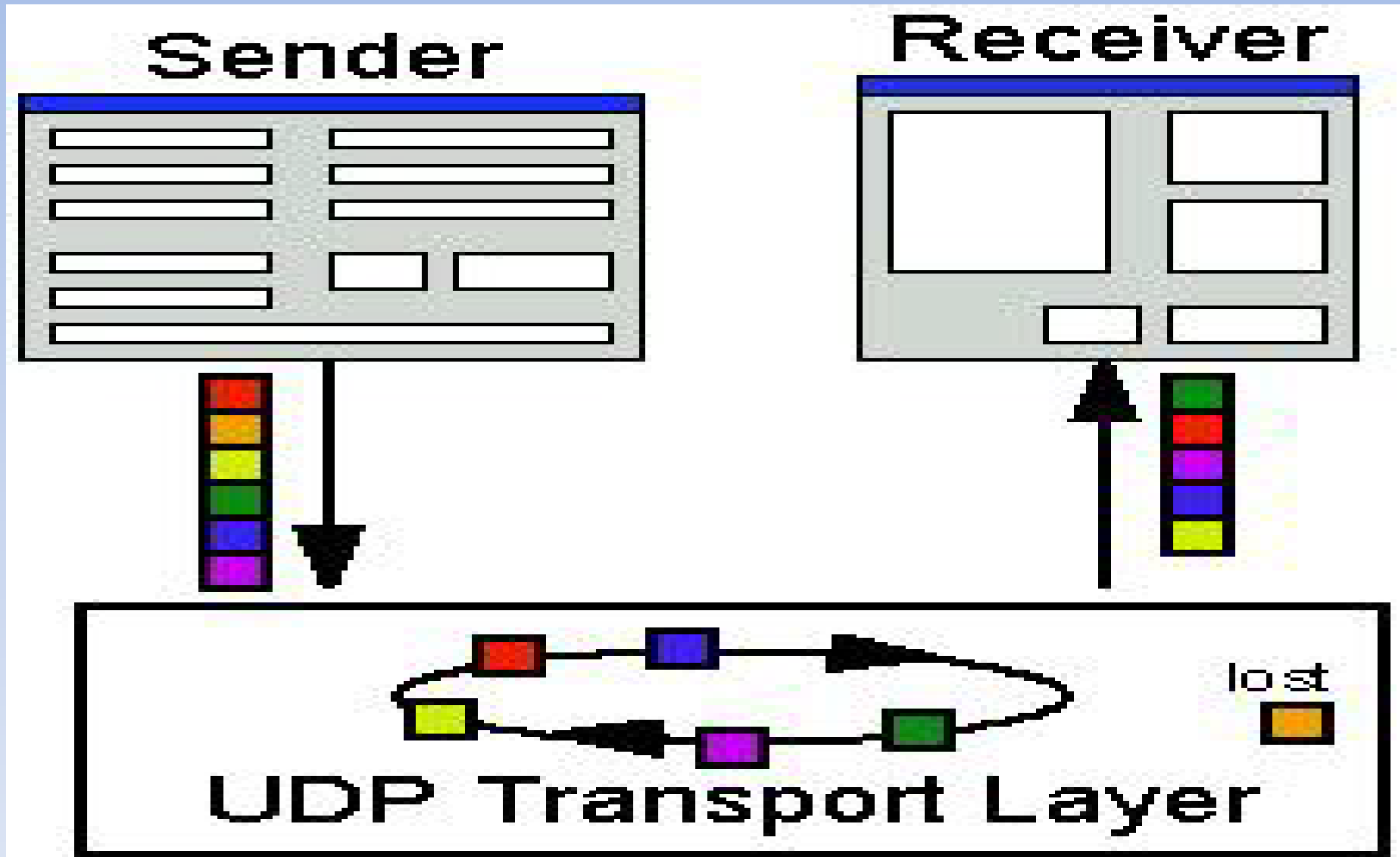
User Datagram Protocol (UDP)

- provides communication that is not **guaranteed** between the two ends
 - sending packets is like sending a letter through the postal service.
 - the order of delivery is not important and not guaranteed.
 - each message is independent of any other.

User Datagram Protocol (UDP)

- **Faster** since no overhead of setting up end-to-end connection
- Many firewalls and routers have been configured NOT TO allow UDP packets.

User Datagram Protocol (UDP)



**Why would
anyone want to
use **UDP** protocol
if information may
get lost ?**



The differences between UDP and TCP

TCP	UDP
Secure	Unsecure
Connection-Oriented	Connectionless
Slow	Fast
Guaranteed transmission	No Guarantee
Used by critical applications	Used by real-time applications
Packet reorder mechanism	No reorder mechanism
Flow control	No flow control
Error Checking	No Error Checkin
20 Bytes Header	8 Bytes Header
Acknowledgement Mechanism	No Acknowledgement
Three-way handshake (SYN, SYN-ACK, ACK)	No handshake
DNS, HTTP, HTTPS, FTP, SMTP, Telnet, SNMP	DNS, DHCP, TFTP, SNMP, RIP, VOIP

A **port** is used as a gateway or "entry point" into an application.



port

- Although a computer usually has a single physical connection to the network, data sent by different applications or delivered to them do so through the use of **ports** configured on the same physical network connection.

port

- When data is to be transmitted over the internet to an application, it requires that we specify the address of the destination computer as well as the application's **port** number.
- A computer's address is a 32-bit IP address.
- The port number is a 16-bit number ranging from 0 to 65,535
 - with ports 0–1023 restricted by well-known applications like HTTP and FTP.

Well-known port assignments

➤ FTP	21	TCP
➤ SSH	22	TCP
➤ telnet	23	TCP
➤ SMTP	25	TCP
➤ HTTP	80	TCP
➤ POP3	110	TCP
➤ IMAP	143	TCP